Paper 11670-2016

Do SAS® High-Performance Statistical Procedures Really Perform "Highly"? A Comparison of HP and Legacy Procedures

Diep T. Nguyen, Patricia Rodríguez de Gil, Anh P. Kellermann, Yan Wang, Jessica Montgomery, Sean Joo, and Jeffrey D. Kromrey

University of South Florida

ABSTRACT

The new SAS high-performance (HP) statistics procedures were developed to respond to the growth of big data and computing capabilities. Although there is some documentation regarding how to use these new procedures, relatively little has been disseminated regarding under what specific conditions users can expect performance improvements. This paper serves as a practical guide to getting started with HP procedures in SAS. The paper describes the differences between key HP procedures (HPGENSELECT, HPLOGISTIC, HPNLMOD, HPREG, HPCORR, and HPSUMMARY) and their legacy counterparts both in terms of capability and performance, with a particular focus on discrepancies in Real Time required to execute. Simulations were conducted to generate data sets that varied on number of observations (10,000, 50,000, 100,000, 500,000, 1,000,000, and 10,000,000) and number of variables (50, 100, 500, 1000) to create these comparisons.

Keywords: HPGENSELECT, HPLMIXED, HPLOGISTIC, HPNLMOD, HPREG, HPCORR, HPIMPUTE, HPSUMMARY, high-performance analytics procedures

INTRODUCTION

Interest in and utilization of 'big data' has exploded in recent years. As researchers and business interests alike are constantly working with bigger and more complex data sets, it comes as no surprise that new procedures are needed. In response to this growing demand, SAS began incorporating high-performance (HP) procedures into available software releases. These changes to existing legacy procedures can decrease overall real time processing by allowing for multiple threads to run concurrently. HP procedures have been designed to work with the existing capacity of the machines on which they are run. They can run in single-machine mode, using multiple threads to decrease run time and can also work in a distributed, cluster environment where multiple threads exists on multiple nodes (computers). Some updated procedures may also improve performance through increased efficiency, with new programming code that is able to take advantage of computing advances made since the legacy code was developed.

This paper provides an introduction to HP statistics procedures (HPGENSELECT, HPLOGISTIC, HPNLMOD, and HPREG) and HP utility procedures (HPCORR and HPSUMMARY). These procedures were contrasted with related legacy procedures to assess possible differences in performance across simulated conditions. As supporting resources for these procedures are still somewhat sparse, the practical purpose of this paper is to help users of traditional SAS procedures become familiar with available HP counterparts and learn under what conditions performance differences of various magnitudes can be expected. These assessments were made using simulated data. The following section describes the steps taken as part of the simulation. The paper then turns to constructing comparisons, detailing the primary distinctions between procedural pairs before addressing differences in performance time. The paper closes with a discussion of results and practical recommendations.

HIGH PERFORMANCE PROCEDURES

There are several benefits associated with the implementation of high performance procedures. Although an exhaustive list is beyond the scope of this paper, the primary benefits have been: allowing for

multithreading, more efficient data steps, and a greater reliance on appliance memory rather than on hard drive capabilities during task execution. For most of these benefits to be realized in full, the HP procedures should be run in distributed mode, meaning that more than one computer (or node) is sharing the task. However, as this paper serves as an introduction to these procedures and many new users are unlikely to be working in a distributed, clustered computing environment, we focus on results from single-machine mode, which is the procedural default.

In single-machine mode only one computer is being used, however this single computer can still have multiple processors (CPUs) running multiple threads or cores simultaneously. Though the benefits associated with revised data steps will not be realized as HP procedures running in single machine mode still access and output data using Base SAS, there should, nonetheless, be differences in running time when contrasting with legacy code due to increases in efficiency and the ability to utilize multiple threads.

Multithreading allows Real Time to decrease because rather than running an entire job sequentially it allows that job to be subdivided into pieces, based on the number of threads. These pieces can then run concurrently. While this sort of procedure can lead to increases in CPU time, as some amount of time is spent splitting up the task, it tends to decrease Real Time due to overlap. However, it is not the case that Real Time will decrease linearly as additional threads are added. A procedure that runs in 30 seconds on one thread is unlikely to run in 15 seconds on two threads. This is due to the increase in CPU time noted above, but is also related to the parallelizability of the process.

Parallelization refers to the amount of a task that can be subdivided among multiple threads. For a portion of a process to be parallelizable, it must be able to run independently of other concurrent tasks. Therefore, when thinking about improvements in executions times, we must be aware of the amount of parallelization that is possible because this will ultimately control the maximum limit on the increase in speed we will observe. While there are several factors related to the amount of parallelization, for the current purpose two are most primary. First, the size of the problem can impact potential speedup. When the problem is small, much of the time associated with the process will be taken up by bringing in the data. As this must be accomplished before other components of the task can begin, this portion is not parallelizable; thus, when bringing in the data takes up the bulk of the process we are unlikely to see significant performance improvements. Conversely, as the size of the problem grows, we expect the fraction of parallelizable code to do the same, thus increasing the amount of speedup. In a similar vein, the shape of the problem can also impact how processing time is allocated. When there are many observations we are again likely to see more time taken by gathering the data; however, when there are a few observations and many variables, there will be more time devoted to computation and thus, a greater percentage of total processing time available for speedup. Although these components will not be addressed directly in this paper, SAS documentation also includes the following components as factors that can impact scalable speedup:

- Options Specified. Some options available for use with HP procedures are not currently coded to allow for multithreading to take place.
- System Load. Background tasks running on the same machine may divert resources from SAS programs and impact potential speedup.
- Hardware. CPU and memory size, speed, and structure will affect scalability. “There are complex interactions between the topology of the CPU arrangements, cache sizes, and user code” (Cohen, 2015).

To the extent possible this study seeks to control for the remaining factors. As such we did not incorporate any of the available options. Additionally, analyses were done on machines not currently running other programs and the same computer was used for each set of procedural pairs.

The simulation used in this study will focus primarily on the impact of problem size and problem shape across pairs of procedures, with pairs being created via a comparison with HP procedures and their legacy analogues. Each pair of procedures will be also run in both personal computers (i.e. desktop or laptop) and cluster computers (i.e. high-performance computers). To the extent possible this study seeks

to control for the remaining factors. As such we will not incorporate any of the available options except for the HP Summary procedure. Additionally, analyses will be done on machines not currently running other programs and specifications of each machine used is provided as a point of reference.

Information on configurations of personal computers used to run the simulation study is provided in Table 1.

Procedure	Configuration			
	Processor	**Memory**	**System type**	**Processing system**
REG & HPREG	Intel I7-3770 MQ CPU, running at 3.40 GHz,	16.0 GB	64-bit	Windows 7 Enterprise
SUMMARY & HPSUMMARY	Intel® Core™ i5-3360M CPU quad core with 2.80GHz each	8.0 GB	64-bit	Windows 7 Professional
NLMOD&HPNLMOD	Intel I5-4570 CPU, running at 3.20 GHz.	8.0 GB	64-bit	Windows 7 Enterprise
GENMOD & HPGENSELECT (Poisson)	Intel® Core™2 Duo CPU E8400 @3.00GHz	4.0 GB	64-bit	Windows 7
CORR & HPCORR	Intel® Core™2 Duo CPU E8400 @3.00GHz	4.0 GB	64-bit	Windows 7 Enterprise
LOGISTIC & HPLOGISTIC	Intel® Xeon ES-2630 @ 2.4 GHZ	32.0 GB	64-bit	LIN X64 Linux
GENMOD & HPGENSELECT	Intel® Core™i3@ 3.1 GHz	4.0 GB	64-bit	Windows 7 Professional

Table 1, Description of Personal Computers Used to Run the Procedures

In addition to running the HP procedures and their legacy counterparts in the personal computers listed in Table 1, we also ran these pairs of procedures via the high-performance computing resources (i.e. Research Computing) administered by the University of South Florida. The High Performance Computing is defined as “consist of hardware and software resources used for computational science that are beyond what is commonly found on the desktop machine” (USF Research Computing website). The Research Computing of University of South Florida hosts a cluster computer which currently contains approximately 476 nodes with processors running Red Hat Enterprise Linux 6. The cluster is built on the condominium model. A 250TB Lustre file system is used to support high speed and I/O intensive computations. The major reason to use the cluster is to speed up the long or multiple instances of computer tasks. SAS UNIX 9.4 platform was used to process the HP procedures and their relevant traditional or legacy ones. This version of the software is installed on Scientific Linux 6.5 compute nodes.

METHODS

Data were generated using PROC IML in SAS 9.4. Sample size (n = 10,000, 50,000, 100,000, 500,000, 1,000,000, and 10,000,000) and number of variables (k = 50, 100, 500, and 1,000) were manipulated to assess the impact of changes in problem size along with problem shape. These factors were fully crossed for most analyses; however procedures involving Poisson outcomes only considered 10 and 20 variables due to convergence issues with the legacy procedures as the number of variables increased. Data for explanatory variables were drawn from normal distributions, while the shape of the distribution for the

outcome variable was altered to allow for testing of the procedures where the outcome is not assumed to be normal (PROC LOGISTIC/HPLOGISTIC and PROC GENMOD/HPGENSELECT).

For each condition, data generation and analysis via the legacy and HP procedures were repeated over 10 iterations. Multiple iterations were selected due to variations noticed in performance time that appeared particular to the unique combination of the generated data set and the procedures being used. To ensure that data presented below were representative of the broader circumstances and were not being driven by a single instance, Real Time and CPU times for each procedure over 10 iterations were averaged. These mean values are reported in the graphs that follow.

As performance time is the focus of this study, results are presented both in terms of CPU Time and Real Time. CPU time is addressed first and then results for differences in Real Time are presented.

RESULTS

Results were discussed with a focus on differences in run time between legacy procedures and related HP procedures in both regular desktops/laptops (PC) and high performance computing cluster (cluster). Sample graphs are provided below, illustrating how Real Time and CPU Time vary between high performance procedures and their traditional counterparts. Similar comparisons will be provided for each condition and each set of procedures in both personal computers and University of South Florida Research Computing cluster when the study is completed.

CPU TIME WITH HIGH COMPUTING CLUSTER

We would expect the variation in CPU time depends on the complexity of the data set. Updates embodied in the HP code improved CPU Time relative to legacy code simply due to the increased efficiency brought on by the new code's ability to better utilize improvements in computing capabilities. At the same time, CPU Time was slightly increased in some cases as the process started with each procedure by allocating portions of the overall task to available threads which took some amount of time, even if very small. Therefore, when the problem is small, we expected CPU Time for HP code to be slightly higher due to increases in time associated with allocation, yet as the data become more complex, HP procedures showed improvements relative to legacy procedures due to increases in efficiency created by the new code.

These expected differences in CPU time were observed from the regression and correlation procedures. When the number of variables ranged from 50 to 500, CPU Time for the HPREG and HPCORR procedures was similar to that time for their legacy counterpart procedures but became higher than the CPU Time in the traditional procedure when sample size is 1,000,000 and larger (Figures 1 and 2).

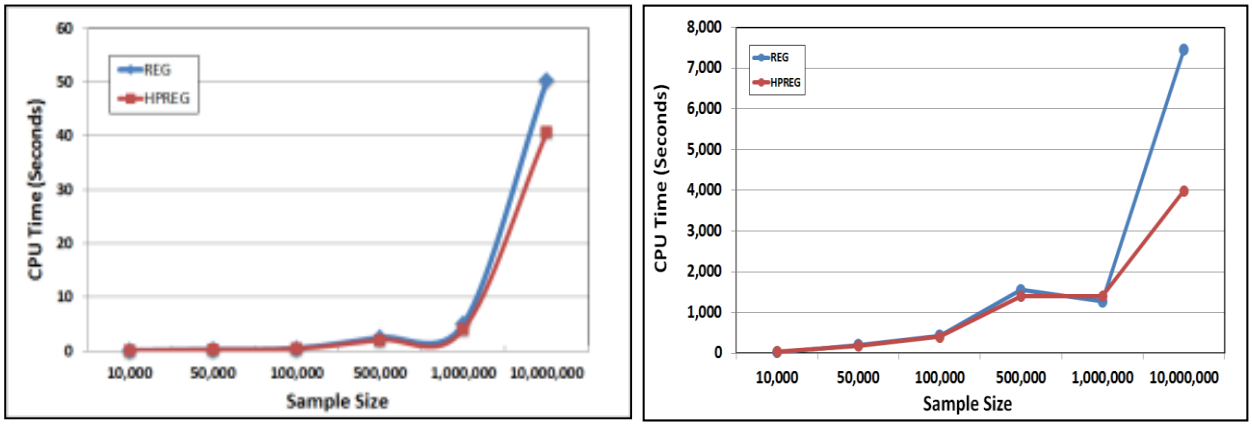
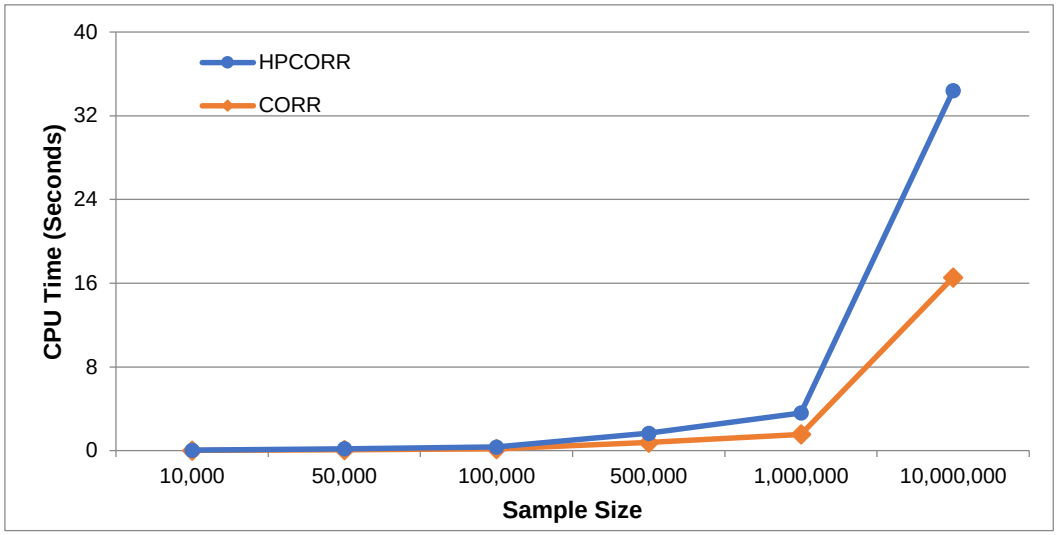


Figure 1, CPU Time for HPREG and REG with k=50 (left) and k=1000 (right)



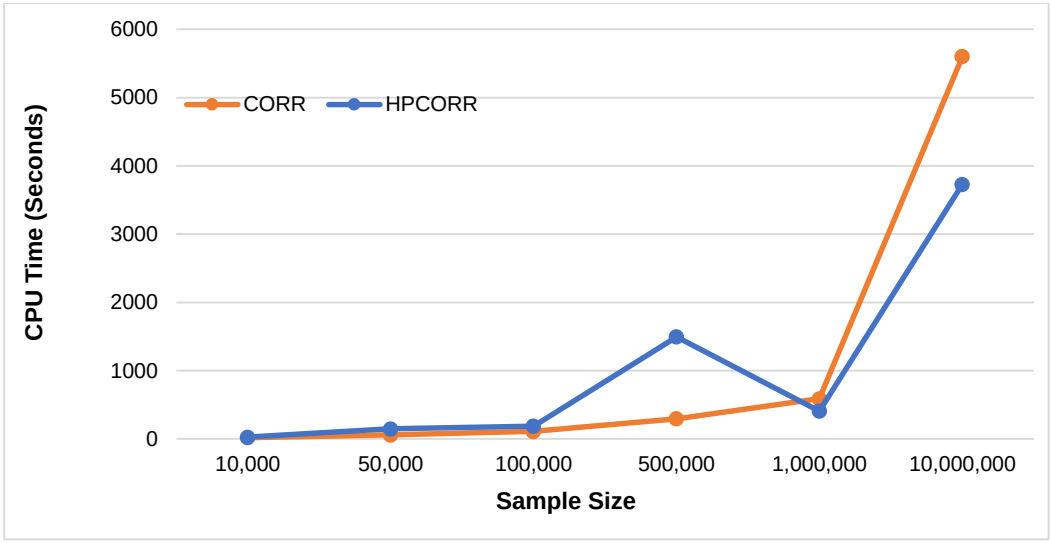


Figure 2, CPU Time for HPCORR and CORR procedures with k=50 (upper) and k=1,000 (lower)

Figure 3 shows that for the HPGENSELECT and GENMOD procedures with normal distribution, CPU time for HP procedure was always shorter than the legacy counterpart across all number of variables from 50 to 1,000 and the time difference between two procedures increases when sample size goes up.

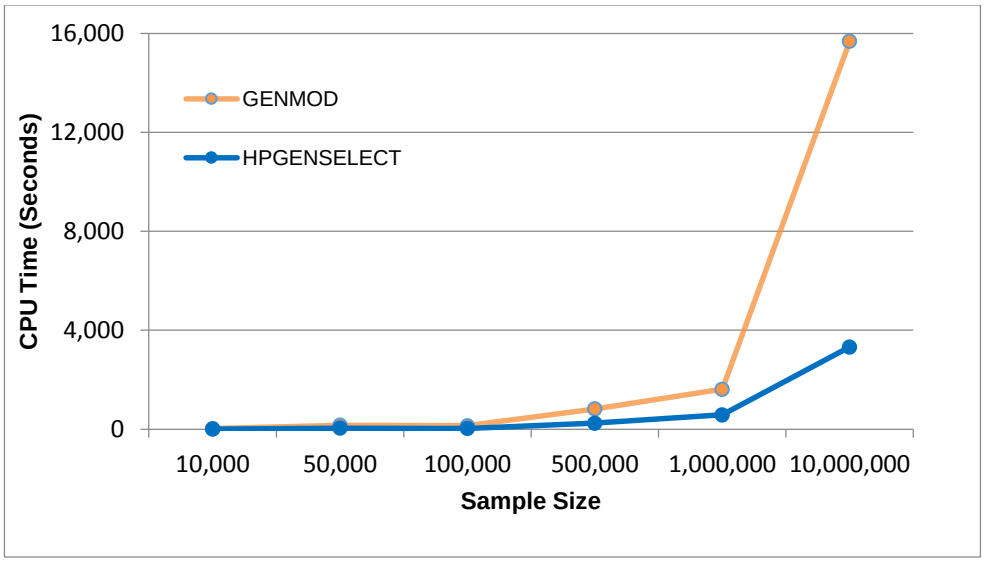


Figure 3, CPU Time for HPGENSELECT and GENMOD with normal distribution (k=500)

This pattern, however, did not hold across all procedures examined. In contrast to the pattern presented in the HPGENSELECT and GENMOD procedures with normal distribution, CPU time for the HPLOGISTIC was always higher than this time for the LOGISTIC procedure across all number of variables. Figure 4 shows the CPU time for these two procedures when number of variables was 500.

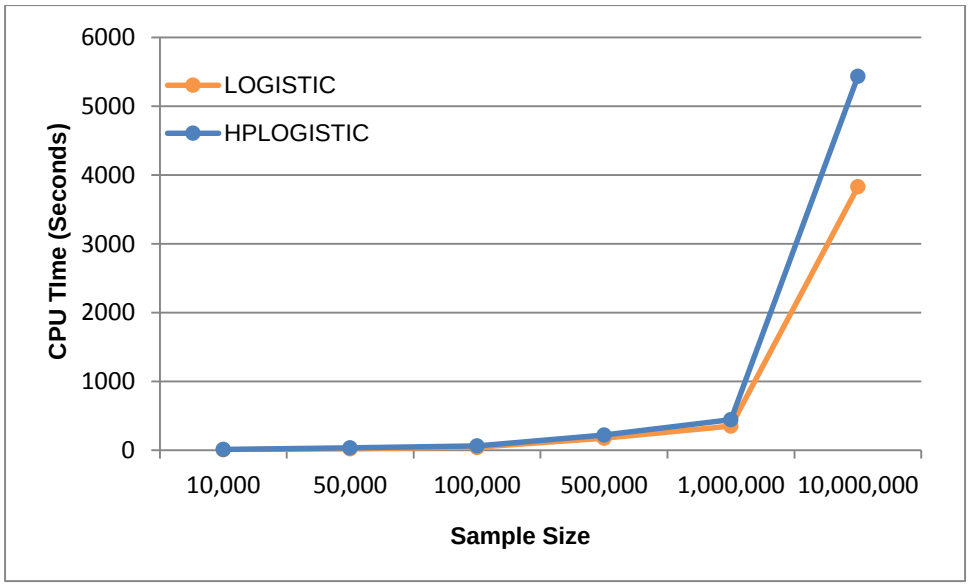


Figure 4, CPU Time for HPLOGISTIC and LOGISTIC (k=500)

In the cases of the HPNLMOD and NLMIXED procedures, since both personal computer and high computing cluster could not run at the smallest condition of variables proposed in this study (i.e. k=50), we also run these procedures with 10 and 20 variables. Figure 5 indicates that CPU time of HP procedure were shorter than those of the legacy procedures across all conditions with k=10 (left) but it was opposite when k was equal to 20 (right).

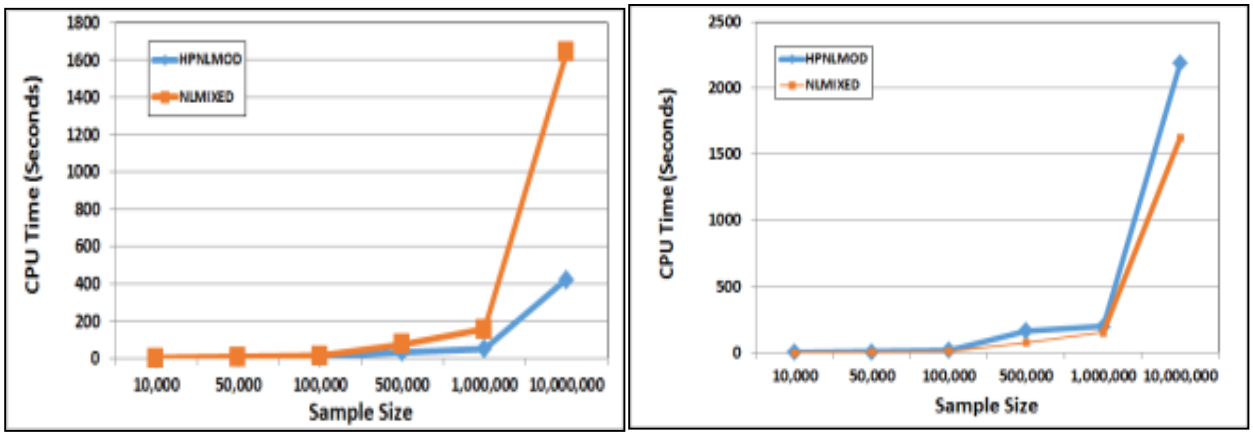


Figure 5, CPU Time for HPNLMOD and NLMIXED with k=10 (left) and k=20 (right)

Only one procedures pair among all procedures investigated in this study shows nearly indistinguishable CPU times across all sample sizes and numbers of variables. With the summary procedures (Figure 6), the amount of CPU time used by the procedures increased practically equally, as the number of observations increased. Here we did not observe any clear differences between CPU Times for the HPSUMMARY and SUMMARY procedures.

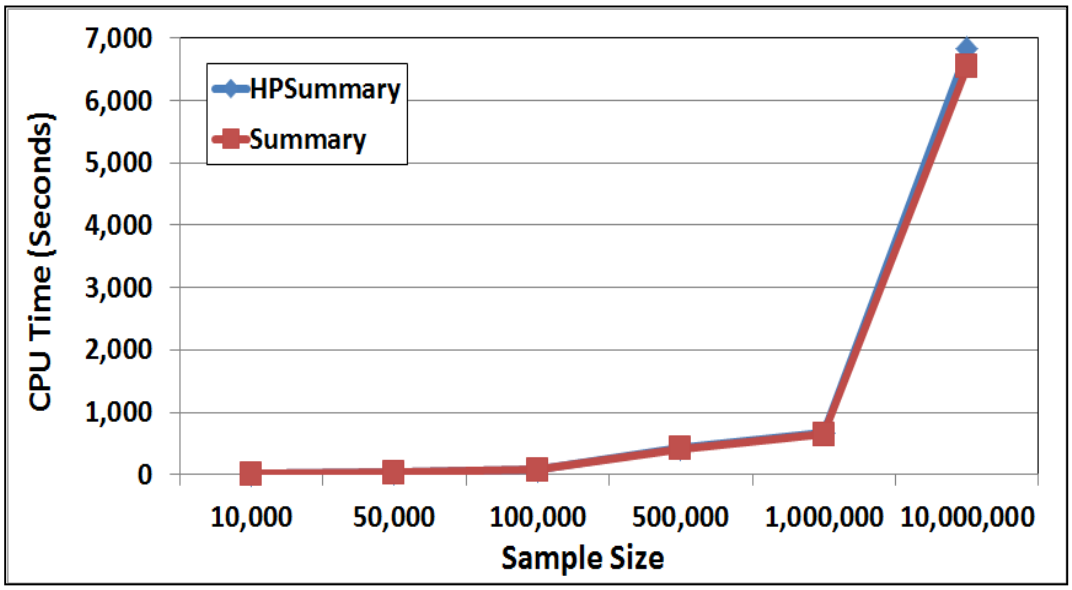


Figure 6, CPU Time for HPSUMMARY and SUMMARY with k=1,000

Although the results of this aspect of the analysis are somewhat mixed, with some procedures evidencing an advantage for HP procedures at larger sample sizes while other suggest the opposite results or no difference at all, one pattern remains consistent. In each of the pairs of procedures analyzed, the HP procedure and corresponding legacy counterpart behaved quite similarly regardless of the number of

explanatory variables included when sample size was below 1,000,000. This suggests that any potential differences are not realized until sample size exceeds a certain bound.

REAL TIME WITH HIGH COMPUTING CLUSTER

We would expect the pattern for Real Time to be similar to that for CPU Time in that the size of the discrepancy between procedures should increase as the problem becomes more complex. We would expect also observe improved performance for the HP procedures here, as a focus on Real Time allows for the benefits (and not just the detriments) of multithreading to be taken into account.

In figures 7 & 8 we observed a similar pattern for these procedures (HPCORR&CORR, HPNLMOD&NLMIXED, and HPGENSELECT&GENMOD with both normal and Poisson distributions) as was observed in the CPU Time section except the HPREG&REG pair. While this pattern is different in CPU time for k=10 and k=20 for HPNLMOD and NLMIXED, it is similar for both conditions in Real Time. Regardless of the number of explanatory variables included, when the number of observations was not large, Real Time averages are similar for pairs of legacy and HP procedures. However, once the sample size surpasses a certain threshold, the difference in Real Time required between the procedures increased significantly. This would suggest that some procedures may have gained more in terms of efficiency, while others only showed improvements in Real Time once multithreading is taken into account.

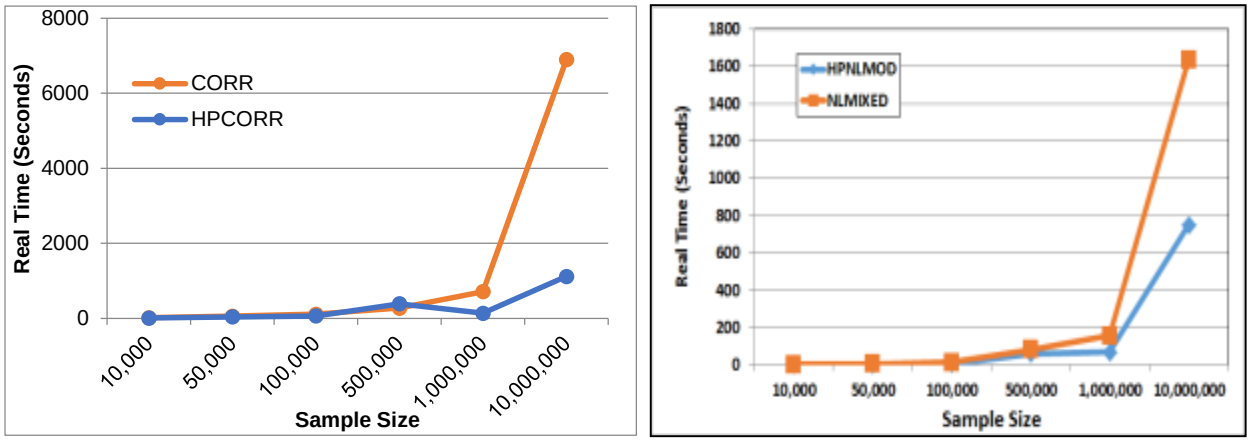


Figure 7, Real Time for HPCORR & CORR with k=1,000 (Left) and HPNLMOD&NLMIXED with k=20 (Right)

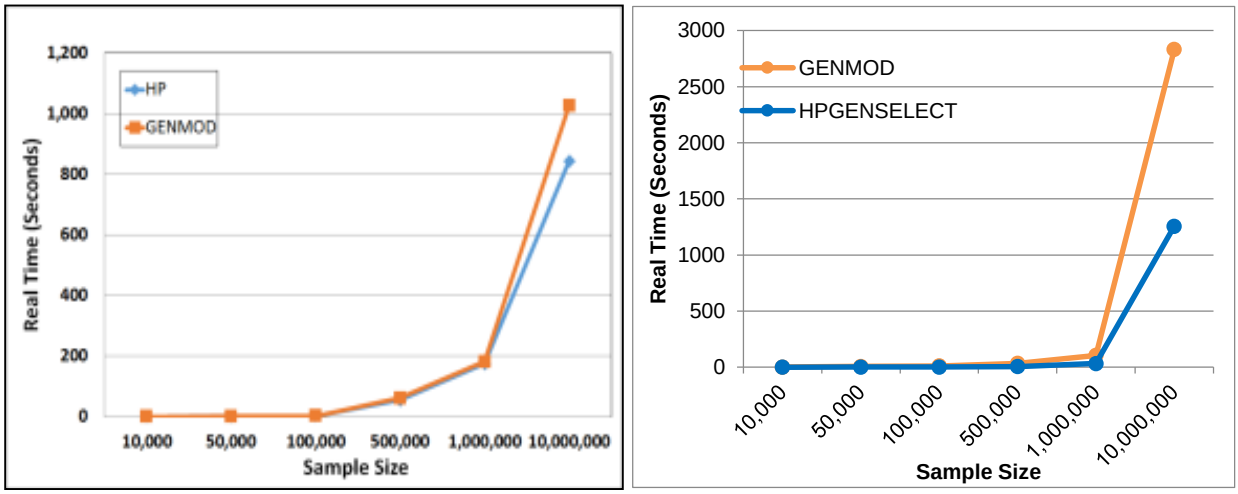


Figure 8, Real Time for HPGENSELECT and GENMOD with k=100 for Poisson distribution (left) and Normal distribution (right)

Unlike its pattern in CPU time for the pair of HPREG and REG but similar to CPU time for the HPLOGISTIC and LOGISTIC procedures, the HP ones in these pairs witnessed longer Real Time than their legacy counterparts (Figure 9).

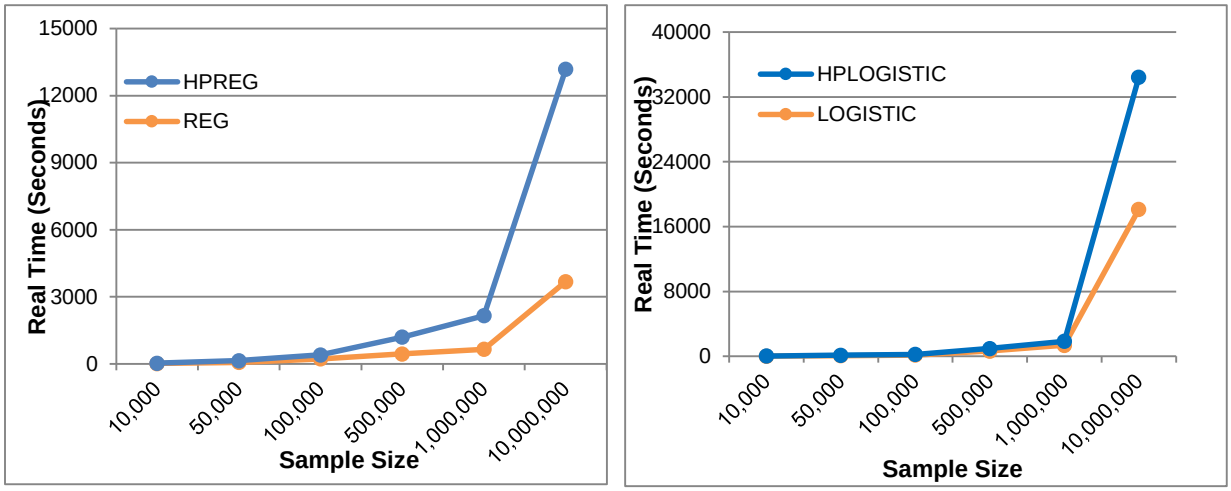


Figure 9, Real Time for HPREG&REG (Left) and HPLOGISTIC&LOGISTIC (Right) with k=1,000

The Real Time for the HPSUMMARY and SUMMARY is similar to what we observed in the CPU time for this pair where there was minimal difference between the two.

CONCLUSION

There was nearly no divergence between CPU Time and Real Time across procedural pairs for the smallest sample sizes in the study (i.e. 10,000, 50,000, and 100,000). It was notable that the legacy procedures performed well even when sample sizes were pretty large and with big number of variables (i.e. 1 million observations and 100 variables). However, as sample size exceeded a certain threshold, greater improvements associated with some HP procedures (HPGENSELECT, HPCORR, HPREG) were observed, suggesting that the efficiency built into this new code provided greater enhancements as the sample size went up. Patterns for Real Time and CPU Time in high computing cluster were relatively similar to what were found in the personal computers.

There were some conditions where some procedures performed well on high computing cluster but could not on personal computers. The pair of HPSUMMARY & SUMMARY worked in cluster but the SUMMARY procedure was shut down when k=500 in combination with N>=100,000 or k=1000 with N>=50,000 in personal computer. The NLMIXED could not perform when k was larger than 20 while the HPNLMOD could still perform in both PC and cluster. The pair of HPGENSELECT and GENMOD with Poisson distribution couldn't run in both PC and cluster when k>100. In that situation, the computer reported that the computer couldn't compute the positive definite covariance matrix.

REFERENCES

What is High Performance Computing. Retrieved January 15, 2016 from the USF Research Computing website:
https://cwa.rc.usf.edu/projects/research-computing/wiki/WhatIsHPC

SAS Institute Inc. 2014. Base SAS ® 9.4 Procedures Guide: High-Performance Procedures, Third Edition. Cary, NC: SAS Institute Inc.
https://cwa.rc.usf.edu/projects/research-computing/wiki/WhatIsHPC

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Diep T. Nguyen
Educational Measurement, Evaluation, and Research program
College of Education
University of South Florida
Tampa, Fl 33620
Phone: 813-974-4933
E-mail: diepnguyen@mail.usf.edu